UCRL-ID-117982

BEGET: The B-Factory Event Generator Version 21

Douglas M. Wright

August 1994

Lawrence Livermore National Laboratory

Ba$\overline{\text{B}}$ar Note # **149**
September 25, 1994
UCRL-ID-117982

BEGET: The B-Factory Event Generator Version 21

Douglas M. Wright
LLNL

Abstract

This note is a reference manual for the B-Factory Event Generator (BEGET V21) software package that generates physics events relevant to B-Factory detector studies. The package provides a standard framework that can easily interface to various external generators and simulation applications. Version 21 of BEGET contains a number of physics and background generators and is interfaced to the JETSET and KORALB generators and the GEANT and ASLUND simulation programs.

Questions and comments should be addressed to wright20@llnl.gov.

This document is also available in hypertext and postscript form:
http://babar1.llnl.gov/wright/doc/beget.html

Contents

List of Tables

1 Introduction

The B-Factory Event Generator (BEGET) provides a simple framework for generating events appropriate for a detector at an asymmetric e^+e^- colliding beam detector. The current version of BEGET, Version 21, can be run in stand-alone mode (for generator level studies) or called from GEANT[1] or ASLUND[2].

BEGET has both built-in generators and interfaces to external generators, such as JETSET[3] and KORALB[4]. For all generators, BEGET handles global tasks such as setting beam energies, smearing the primary vertex, and boosting to the final (beam asymmetric) reference frame. It also has a number of features to facilitate CP asymmetry studies. For example, there are switches to activate the rare B decay modes to CP eigenstates and routines that simulate CP violation and mixing.

Calling BEGET produces a single event stored in a FORTRAN common block in a format defined by the STDHEP software package[5]. The common block contains a list of the generated particles with their associated four-vectors and vertices. STDHEP defines not only the common block but also the particle code convention, so that the interface to any external simulation package is handled by a single subroutine that reads the common block.

BEGET is controlled by an ASCII file containing keywords and parameters, called 'data cards.' The data cards are defined and read in with the FFREAD[6] package from the CERN library. Except for some of the beam related background generators, no external data files are required to run the program. All defaults are built-in; however, the user can override nearly every aspect of the program through data cards and external files. For example, the decay table for selected particles (or all particles) can be replaced.

The generators currently available in BEGET are:

1. Single particle. The user specifies the particle and four-vector range.
2. $\Upsilon(4S)$ decay via JETSET.
3. Hadronic continuum events from JETSET (i.e. $e^+e^- \rightarrow q\bar{q}$).
4. Quark "Onia" resonance decays via JETSET.
5. KORALB $\tau^+\tau^-$ decays.
6. Beam background from off-momentum particles.
7. Background from beam-gas interactions.
8. Cosmic rays.

Additional generators and decay packages (e.g. QQ from CLEO) are in the works, and users are encouraged to interface their own favorite generators.

1.1 Υ(4S) and B^0 Decay

The current Υ(4S) generator uses JETSET and is a modified version of what is found in the native ASLUND generator. A proper treatment of Υ(4S) production that includes the upsilon width has not yet been implemented. The Υ(4S) decays into an equal mixture of $B^0\bar{B}^0$ and B^+B^- final states.

The standard JETSET decay tables are modified with updated (circa 1992) heavy meson decay data (B and D) from CLEO and ARGUS. See subroutines `BEGFILCST` and `BDECAY` for more details. I have added the B^0 decay modes of interest for CP violation studies which are listed in Table 1.[1]

Table 1: Decay modes of interest for CP violation.

B^0 Decay Mode	BR ($\times 10^{-4}$)
$\psi\ K_L^0$	3.85
$\psi\ K_S^0$	3.85
$\psi\ K^{*0}$	12.60
$D^+\ D^-$	6.00
$D^{*+}D^{*-}$	15.00
$\pi^+\ \pi^-$	0.20
$\rho^\pm\ \pi^\mp$	0.80
$a_1^\pm\ \pi^\mp$	0.80

A built-in switch allows all B^0 (or $\bar{B}^0$) decay modes to be turned off except for a selected mode. For example, to generate the following decay:

$$\Upsilon(4S) \rightarrow (B^0 \rightarrow \psi K_L^0)\ (\bar{B}^0 \rightarrow \mu^- \bar{\nu}_\mu X)$$

One would use the following data card: `BOBO 11 3`. The keyword is `BOBO`, the first number specifies the B^0 decay ($11 = \psi\ K_L^0$) and the second specifies the $\bar{B}^0$ decay ($3 = \mu^-\ \bar{\nu}_\mu X$). Instead of using the built-in switches the user can also alter the B^0 decay, or any other decay, with an external file that contains the desired changes to the decay table (data card JETD).

By default, CP violation and B^0 mixing is turned off. The user can activate it with data card CP. In the current implementation, the CP part only works on pure CP eigenstate modes. There is a also an option in JETSET to do B^0 mixing. Currently, the user must activate this by hand (i.e. by setting the value in the appropriate JETSET common block).

Section 5 on page 14 contains a complete description of the available data cards.

1.2 Acknowledgments

Acknowledgments traditionally appear at the end of a paper but are included where the reader is more likely to seem them.

[1]The branching ratios match those in the BaBar LOI [7].

BEGET began as a synthesis of old and new subroutines. The old was taken primarily from ASLUND and the only credit that I can ascribe comes from comments left in the source code itself. The ASLUND simulation began as a modification of another fast, parameterized simulation. Alan Weinstein created the original ASLUND and Art Snyder maintains and develops the current version. Some of the JETSET modifications and utilities should be attributed to Patrick Roudeau and Jon Guy. In particular the heavy meson decay modifications come from them. The routines that simulate CP asymmetry are due to Alan Weinstein with modifications from Gautier Hamel de Monchénault, Dave Coupal, and Walt Innes.

A number of people have contributed to BEGET since its first release (Version 0, January 1994). In particular Fred Kral has implemented the KORALB interface, the beam-gas interaction interface, and a cosmic ray generator (with Mike Bringle). Ezio Torassa contributed a routine for picking background rays from TURTLE files prepared by Art Snyder. Dave Coupal has made extensive changes to this routine.

Contributions from others are soon to be incorporated and as the program matures I hope that this section continues to grow.

2 Using BEGET

A single event is generated by calling subroutine `BEGET` (i.e. `CALL BEGET`). This fills the STDHEP common block with the final state particles and the intermediate particles that decayed. Subsequent calls to the `BEGET` subroutine put a new event into the common block.

The general purpose part of BEGET must be initialized before the first call to the `BEGET` subroutine. This initialization is handled by a different subroutine that depends on whether one is running BEGET stand-alone or from GEANT or ASLUND. The main purpose of the initialization is to read in the data cards that define what to generate.

The proper calling structure is already built-in to ASLUND and BBSIM,[2] so users of those programs do not need to modify their code. The initialization for a particular generator package (e.g. JETSET) is automatically called in the first pass through the BEGET subroutine, so no special user action is required. Be aware that BEGET is not currently configured to allow input parameters to change between generated events.

2.1 Using BEGET for Generator Studies

The following two program examples can be found in the `example` directory of the source code distribution. Shell scripts `beget1` and `beget2` contain the proper compilation and linking commands.

To make a stand-alone version of BEGET, use a main program like the following (which generates 10 events).[3]

```
program main
call beginput ! define and read in data cards
do n = 1, 10
```

[2]BBSIM = B-Factory GEANT detector simulation.

[3]Unlike BBSIM, users do not need to copy any of the BEGET source code.

```
      call beget ! generate an event
   enddo
   end
```

Link the program using the following command,

```
f77bb -link -o beget1.exe beget1.f `$BFLIB/beget/Link`
```

where the main program has been saved in file `beget1.f`. The back quotes (`) are important, they cause the output of the `Link` script to be included on the command line. The `Link` script produces the appropriate list of libraries needed to link BEGET. Linking is discussed in detail in Section 2.6.

This small example results in a complete, functioning version of BEGET. It can be used to familiarize oneself with the function of the various BEGET data cards.

For generator studies one needs access to the STDHEP common block, which is defined in the include file `stdhep.inc`.[4] A slightly more complicated main program that illustrates how one might read the common blocks is given below.

```
      program main
#include "stdhep.inc"
#include "beget.inc"

      parameter( ID_PI0 = 111 )
      call beginput              ! define and read in data cards
      do n = 1, Nbeg_events      ! Nbeg_events is defined by EVNT data card
         call beget              ! generate an event

*........analyze this event
         Npi0 = 0
         do i=1, NHEP            ! loop over particles in the event
            if( IDHEP(i) .eq. ID_PI0 ) Npi0 = Npi0 + 1
         enddo
         print *,'There are ',Npi0,' pi0''s in this event'
      enddo
      end
```

This example merely counts the number of π^0's in the current event by looping over all of the particles and checking the particle code in array `IDHEP`. Section 6 contains a summary of all of the variables defined in the STDHEP common block.

The BEGET internal common block, defined in `beget.inc`, gives the user access to the values of the input data cards and other internal BEGET variables. To compile and link this program, one must specify the location of the includes files. The following commands will do the trick:

```
f77bb -cpp "-I$BFLIB/beget -I$BFROOT/src/stdhep/pro" beget2.F
f77bb -link -o beget2.exe beget2.o '$BFLIB/beget/Link'
```

[4]The include file is stored with the source code which is typically kept in `$BFROOT/src/stdhep/pro`.

The above examples expect that the `f77bb` shell script is used for the compilation and linking of the program, however, this is not a requirement. The `Link` script is general enough to work with any UNIX linker and the main program will work with a generic FORTRAN compiler if the C style includes (`#include`) are changed to FORTRAN style includes.

2.2 Using BEGET in GEANT

To use BEGET with GEANT[5] make a GUKINE subroutine like the following.

```
subroutine gukine
call beget_geant ! generate an event
end
```

and make sure that the following call is made before the call to GFFGO.

```
call begffkey   ! define BEGET FFREAD data cards
```

The GEANT implementation works by calling the `BEGET` subroutine and then subroutine `HEP2GEANT` to load the contents of the STDHEP common block into the GEANT KINE and VERTX banks.

Link your GEANT program with `` `$BFLIB/beget/Link` `` as an argument to your link command (note the ``'s are important). For more information on linking see Section 2.6.

Some BEGET routines communicate with GEANT through the standard GEANT common blocks. The current version of BEGET is compiled with the GEANT 315 include files from CERNLIB version 94a. If you want to use BEGET with any other version of GEANT you should recompile the necessary routines. For easy reference, these routines are kept in the `$BFLIB/beget` directory. This is done automatically for BBSIM users.

2.3 Using BEGET in ASLUND

The next release of BEGET will become the default generator in ASLUND and the interface is likely to change at that time.

The BEGET interface is already built into ASLUND, however, it is not linked into the ASLUND executable by default. To link with BEGET you must uncomment the line

```
#define BEGET
```

in the ASLUND `user/Imakefile` and remake your executable. You may have to copy the most recent version of `Imakefile` to access the most recent version of BEGET, since the link syntax changed between BEGET version 20 and 21.

Once you have a version of ASLUND that contains BEGET, set `IFL=-99` in the `runtime` input file, this turns off the built-in generator and turns on BEGET. BEGET data cards should be embedded in the `runtime` file, they are not read in separately. For ASLUND to recognize the BEGET data cards `;BEGETBEGIN` must appear on a separate line before the first card and `;BEGETEND` must follow on a separate line after the last card, For example:

[5]BBSIM users: BEGET is already implemented in BBSIM, so there is no need to change any source code.

```
;BEGETBEGIN
BOBO 3 11
GENP 1
;BEGETEND
```

This only applies to ASLUND. The stand-alone and GEANT version of BEGET do not require `;BEGETBEGIN, ;BEGETEND`.

The default beam energies are taken from ASLUND variables instead of BEGET, but the defaults can be overridden by the BEGET data cards BEAM and SBEAM. Also, ASLUND does its own vertex smearing so data cards VERT, SVERT, IVERT should not be used at this time.

ASLUND is not built around STDHEP, it looks directly at the JETSET common blocks to find the generated particles. The BEGET interface to ASLUND makes use of a STDHEP translation routine that converts STDHEP to JETSET. So non-JETSET generators (in BEGET) are automatically available in ASLUND.

2.3.1 Comments on Particle Codes for ASLUND

Some ASLUND extensions to the JETSET particle code conventions are not compatible with STDHEP (i.e. particle codes are used that either map incorrectly into the STDHEP scheme or do not exist at all). In general this is not a problem. Users who modify the JETSET decay tables with an external file may run into trouble if they try to use the same file with BEGET. However, by following a few guidelines an external file can be made that works with both BEGET and the ASLUND native generator.

Traditionally ASLUND uses 40553 as the code for Υ(4S), which is not defined by JETSET, but is defined by STDHEP as code 70553. The STDHEP convention for $b\bar{b}$ resonances is given in Table 2.

Table 2: Particle Code Convention for $b\bar{b}$ resonances.

	Particle Code	
	ASLUND	STDHEP
$h_b(2P)$		40553
$\chi_{b1}(2P)$		50553
Υ(3S)		60553
Υ(4S)	40553	70553
Υ(5S)	50553	80553

The ASLUND user may select particle 70553 as Υ(4S) but must be careful to use the correct "compressed" code associated with it. Particle codes in JETSET are "compressed", i.e. mapped to a contiguous range of numbers 1-500. The function `LUCOMP(kf)` returns the compressed code for particle code `kf`. JETSET does not define the mapping for particles with compressed codes 400-500. It is up to the user to define the mapping.

The ASLUND version of `LUCOMP` has a mapping given in Table reflucomp. In ASLUND

Table 3: ASLUND version of LUCOMP mapping.

KF	LUCOMP(KF)
40553	401
50553	402
60553	403
70553	404
80553	405
nXXXXXX	400 + abs(n)

there are (at least) two ways to get a compressed code of 404. For example 70553 and 4000511 both compress to 404. To make decay table files that are compatible with both the native ASLUND generator and BEGET, use code 70553 and compressed code 404 for $\Upsilon(4S)$, and make sure that any other user particles give a compressed code greater than 410.

The stand-alone and GEANT version of BEGET do not respect the `LUCOMP` mapping of ASLUND for particle codes `nXXXXXX` (i.e. $> 99,999$) because of the possible inconsistencies stated above. So decay table files can be shared between the ASLUND native generator and BEGET only within ASLUND. With sufficient user interest, the discrepancies in the `LUCOMP` mapping could be eliminated.

2.4 Generating Background

Three types of background events can be generated:

1. Off-momentum beam particles from previously prepared TURTLE files
2. Single hadrons from Beam-Gas interactions
3. Cosmic rays

At present, only the first background (off-momentum particles) can be overlaid with other physics generators.

2.4.1 TURTLE Interface

(Code and documentation provided by Dave Coupal.)

Overview

BEAM-GAS TURTLE is used to generate a file of Coulomb and bremsstrahlung scattered beam particles (or brem photons) that hit near the interaction point. A background event can contain one particle (as in Coulomb scattering) or two if both a brem-scattered beam particle and its associated photon strike near the interaction point. The position of the scattered particle at the entrance to quadrapole Q1 is input to BBSIM and GEANT then tracks the particles to the point of impact with a beamline component and simulates the resulting electromagnetic shower. These back- ground events can be examined as events by themselves or overlaid on a physics event.

The input of background events to BEGET is controlled by the `BACK` and `BACF` data cards.

Recommended Use

There are two recommended methods of using the TURTLE ray input. Ideally one should, for a given GEANT geometry, run through the files of background events, one event at a time, and generate output events of GEANT hits that could then be overlaid with events of interest in a subsequent analysis stage (with the timing of the hits smeared over the desired time window). This way the time-consuming step of running GEANT on the file of background events is performed only once (for a given geometry). To do this one would set the `BACK` card to:

```
BACK 1 0.  0.  1.  (with no GEN card)
```

To calculate the time represented by the number of ray-events run:

```
total time = ( 1. / (sum of prob/crossing)) * (# of ray-events read)
                         *  (time/crossing = 4.2 nsec)
```

(This number is printed at the end of the job.)

An alternative depending on your patience is to input the events on top of a physics event at the event generator level (so GEANT has to shower the particles for each event). In this case one would use something like:

```
GEN 2
BACK 1 -119 118 0.
```

This would generate an $\Upsilon(4S) \rightarrow B^0\ \bar{B}^0$ and add in background events over a range of beam crossings from -119 to 118 (for a +- .5 microsecond window, (238 crossings/microsecond)). This assumes the detector simulation code is set up to use the timing information associated with the tracks to generate the hits or digitizations.

[With some CAUTION one can, for certain studies, use the imode = 2 option to rewind at End-of-File and continue using the same events over again.]

2.4.2 EPC: Beam-Gas Background

(Code and documentation provided by Fred Kral.)

EPC is a stand-alone program used by Lew Keller and Fred Kral to generate tables of double-differential cross-sections for electron-gas scattering with single-particle final states such as protons.

The BEGET interface reads EPC ascii output cross section files and produces single hadron final states. The files are expected to be in the current working directory, so the user should make links with the `epclinks` script (found in the BEGET `dat` directory). The user may need to edit this file. Use data card `GEN 11` to turn on this generator. Note that you cannot mix this background with other generators at this time.

EPC Delta production Protons

EPC gives rise only to single-particle states. This is a way to estimate cross-sections of the two-particle Delta production final state: Pions are produced only in `Delta -->`

`nucleon pion`. To get the protons from this reaction, the program was kludged to only use the Delta part. The files `epc-N9p-delta.dat` and `epc-N3p-delta.dat` contain cross sections for protons produced exclusively through Delta decay. To turn off the remaining 8 files, a blank file was created, `epc-blank.dat`. The script `epcdelta` over-rides the usual links with the blank files and the two proton delta files.

2.4.3 Gosmic Ray Background

(Code and documentation provided by Fred Kral and Mike Bringle.)

Cosmic rays that reach a detector cylinder (default parameters to current BBSIM mother volume) can be generated with data card `GEN 12`. Data card `COSM` allows the user to control the cosmic ray spectrum and detector volume. The generator expects a number of input files to be present in the current working directory. The shell script `cosmlinks` can assist the user in making links to these files.

2.5 User Hooks and Utilities

A few hooks are provided to allow a more detailed control of BEGET. The following subroutines are dummies in the BEGET library. The user may write his/her own subroutines and link them before the BEGET libraries.

- Subroutine `BEGUSER` is called instead of any other generator, if `GEN -99`. It provides a means for users to write their own generators or develop a new interface to an external generator.
- Subroutine `BEGUNIT` is called after the requested generator has been initialized, but before any events are generated, to give the user an opportunity to change the initialization of the generator.
- Logical Function `BEGUREJECT` allows the user to reject the event and request a new one before returning from the `BEGET` subroutine.

None of the subroutines or functions take any arguments.
Some BEGET subroutines that may be of interest to users are:

- `HEPROBO` performs Lorentz boosts on all of the particles in the STDHEP common block.
- `BEGHBOOK` initializes the HBOOK histogramming package.
- `HEP2GEANT` reads the STDHEP common block and places the undecayed particles in the GEANT KINE and VERT banks.

Please refer to the source files in the `src` directory of the BEGET distribution for more information.

2.6 Linking BEGET

Since a number of libraries and object modules are needed to link BEGET with your program a shell script (`Link`) is provided to produce the proper linker arguments. This not only simplifies linking but also allows the easy switching of BEGET versions and the selective linking of external generators (such as KORALB).

The script produces a text string that contains the actual linker arguments in the appropriate order. The standard Unix trick of using \` permits the output of the script to be included as part of another command, e.g. `` f77bb -link test.o `Link` ``. To see the output of the script enter it as a regular UNIX command without the \`'s.

The BEGET version number is hardwired into the script so that it always links a specific version (even if the script is accessed via a link). The script looks for the beget libraries in the same directory in which the script resides (e.g. `$BFLIB/beget/Link` looks in `$BFLIB/beget`). In addition the script requires two environment variables `$BFLIB` and `$CERN_ROOT`, the first is needed to locate the STDHEP library and the second is used to locate the CERN libraries.

A few examples:

`` `$BFLIB/beget/Link` ``	Link BEGET default version.
`` `$BFLIB/beget.v021/Link` ``	Link BEGET version 21.
`` `$HOME/beget/Link` ``	Link BEGET version found in the user's home directory.

2.6.1 Selective Linking of Generators

By default BEGET links the JETSET library. JETSET is required for the random track generators (to do the particle decays if the user requested a non-stable particle) and the $\Upsilon(4S)$, $q\bar{q}$, and quark "Onium" generators. Other external generators are not linked automatically. This keeps the resulting executable small, minimizes linking time and eliminates the need for every site to install every generator.

Currently, the only external generator available is KORALB. To activate it you must install the KORALB library on your system (a development version is already installed at SLAC) and add an argument to the linker script to tell BEGET where it is.

For example if KORALB is installed as `libkoralb.vdev.a` in the `$BFLIB` directory then

```
`$(BFLIB)/libbeget.Link -lkoralb.vdev`
```

would produce a BEGET program that can access KORALB. ASLUND users must edit a line in the ASLUND `user/Imakefile` to look like the following:

```
BEGETLIBS =`$(BFLIB)/libbeget.Link aslund -lkoralb.vdev`
```

To actually generate KORALB events the user must select the `GEN 10` data card at run time.

New external generators (for example QQ) will be made available via the same mechanism. They will not be linked automatically unless the user requests them.

3 Installing BEGET

Before installing BEGET, you must install the STDHEP library (`libstdhep.a`) and include file (`stdhep.inc`). This is not part of the BEGET distribution and is not yet available via CVS. The STDHEP source code is typically kept in `$BFROOT/src/stdhep/pro`. The STDHEP source code will be in CVS by the time of the next BEGET release.

The master copy of BEGET is stored on the SLAC Unix cluster in CVS. There are three method to install it at a remote site:

1. Copy the binaries and include files.
2. Copy the 'tar' file and compile BEGET.
3. Check out the CVS version and compile BEGET.

The third method has the advantage that bug fixes can be automatically incorporated at the remote site.

BEGET has been successfully installed under the following operating systems: HP-UX, AIX, SunOS, Alpha/OSF, and Ultrix. Any problems with these or other platforms should be reported to `wright20@llnl.gov`.

3.1 Method 1 (Copying the Binaries)

Copy the entire contents of the $BFLIB/beget directory, from a site that has already installed BEGET, of course the CPU type must match your system.[6]

3.2 Method 2 and 3 (Compiling BEGET)

Installing BEGET from the source code is a three step process:

1. Copy the source code to your system from either

   - the most recent tar file in `$BFROOT/src/beget`
   - via CVS: `cvs -R co -r v021 -d v021 beget`

   Store the source code in `$BFROOT/src/beget/v021` and maintain a `pro` link to the current version, which in this case is `v021`.

2. Compile the code. Type `gmake` in the BEGET directory you just created. This builds the libraries in the current directory. You may have to edit the `GNUmakefile` to specify the location of the STDHEP include file, i.e. edit the line:

   ```
   hepinc := $(BFROOT)/src/stdhep/pro
   ```

3. Move the libraries to the `$BFLIB` area. Type `gmake install` to move the libraries from the working directory to the `$BFLIB` area.

[6]The SLAC Unix cluster has the IBM RS6000 version ($BFROOT/lib.aix6000) and the HP version ($BFROOT/lib.hp).

3.3 Software Requirements

Table 4 outlines the software required toinstall BEGET which does not come with the BEGET distribution.

Table 4: BEGET software requirements

Required Environment Variables:	
BFLIB	final location of BEGET libraries
CERN_ROOT	location of cern libraries (e.g. /cern/pro)
Required to install BEGET:	
f77bb	shell script for compiling FORTRAN
libstdhep.a	STDHEP library
stdhep.inc	include file for STDHEP common block
libpacklib.a	CERN library
libmathlib.a	CERN library
libjetset.a	CERN library
Optional	
libkoralb.a	KORALB library

4 Caveats

Caveats on running BEGET.

- When linked to GEANT the following particles: K_S^0, $\Sigma^{\pm 0}$, Ξ^{-0}, Λ^0, Ω^-, are declared to be stable (which JETSET normally decays). This allows them to be passed on to GEANT which then handles their decay.

- When linked to a GEANT version other than 315 from CERNLIB 94a, you should recompile and link the subroutines found in the `$BFLIb/beget` directory. This is done automatically in BBSIM.

- The $\Upsilon(4S)$ width is not used when beam smearing is activated.

- KORALB does not handle smearing of beam energies. This would require reinitializing KORALB at every event, which has not been implemented.

- When linked to ASLUND, the default beam energies are taken from ASLUND variables instead of BEGET, but the defaults can be overridden by the BEGET data cards BEAM and SBEAM.

- When running ASLUND do not use data cards VERT, SVERT, or IVERT, since ASLUND does its own vertex smearing after the event has been generated.

- When linked to ASLUND, BEGET uses the ASLUND version of subroutine `LUCOMP` instead of its own version. The mapping of user defined particles is different in the two cases. Power users who exploit the mapping of these non-standard particles should review Section 2.3.1.

  Carefully constructed decay table files can be used with both the ASLUND native generator or BEGET. The stand-alone and GEANT version of BEGET use a different version of `LUCOMP` which is more robust. The next version of BEGET will resolve this discrepancy

5 Controlling BEGET

This section describes the data cards used to control BEGET. The data cards are read in with the FFREAD package from the CERN library. The defaults are set in subroutine `BEGFFKEY` and `BEGINIT`.

5.1 Data Card Overview

Data Card	Function
	Generator Controls
GEN	Choose what to generate
CP	Select CP violation and/or B^0-$\bar{B}^0$ mixing
BACK	Activate beam background from TURTLE files
BACF	Set the location of a TURTLE file list
COSM	Control cosmic ray generator
	Decay Controls
B0B0	Selectively deactivate B^0 and/or $\bar{B}^0$ decay modes
JPSI	Control J/ψ decay
JETD	Specify a file to read in JETSET decay information
	Beam Controls
BEAM	Set the beam energies
SBEAM	Set the beam energy spread
VERT	Set the primary vertex location
SVERT	Set the primary vertex spread
IVERT	Set the primary vertex smearing method
	General Controls
GENP	Print event information on the terminal
JETO	Print the decay table for the listed particles
RAN	Control the random number seeds
EVNT	Set the number of events to generate
REJE	Control user event rejection

5.2 Data Card Default Values

Data Card	Default Value	Default Function
		Generator Controls
GEN	2	Generate $\Upsilon(4S) \rightarrow B^0\ \bar{B}^0$, decay using JETSET
CP	0	No CP violation or B^0-$\bar{B}^0$ mixing
BACK	0	do not add TURTLE background
BACF	”	no default TURTLE file list
COSM	(see below)	default BaBar geometry used if cosmics are generated
		Decay Controls
B0B0	1 1	Turn on all B^0 decays
JPSI	-1	Turn on all J/ψ decays, unless `B0B0` is not 1 1 (see below)
JETD	”	Do not read any external decay file
		Beam Controls
BEAM	9.0 -3.109	$p_z(e^-)$ = 9.0 GeV/c, $p_z(e^+)$ = -3.109 GeV/c
SBEAM	0. 0.	zero energy spread on beam (no smearing)
VERT	0. 0. 0.	event origin is (0,0,0)
SVERT	0. 0. 0.	no vertex smearing
IVERT	0 0 0	use gaussians (sigmas set by SVERT)
		General Controls
GENP	0 0	Do not print the events on the terminal.
JETO	0*20	no print out of decay tables
RAN	1 0	use default seed #1 = 9876 54321
EVNT	1	set `Nbeg_events = 1`
REJE	0	accept all events

5.3 Data Card Definitions

GEN: Choose what to generate

```
*****************************************************************************
GEN   IMODE KF x1 x2 x3 x4 x5 x6                      Choose what to generate
*****************************************************************************

GEN       2  0  0. 0. 0. 0. 0. 0.          Defaults

       OVERVIEW
       +----------------------------------------------------------------+
       |-100    NULL Physics Generator                                  |
       | -99    call BEGUSER                                            |
       +----------------------------------------------------------------+
       |       Single Particle (Pt)                                     |
       | -13 kf Ptmin Ptmax cos(themin) cos(themax) phimin phimax       |
       | -12 kf Ptmin Ptmax themin themax phimin phimax                 |
       | -11 kf Pt cos(theta) phi                                       |
       | -10 kf Pt theta phi                                            |
       +----------------------------------------------------------------+
       |       Single Particle (P)                                      |
       | -3 kf Pmin Pmax cos(themin) cos(themax) phimin phimax          |
       | -2 kf Pmin Pmax themin themax phimin phimax                    |
       | -1 kf P cos(theta) phi                                         |
       |  0 kf P theta phi                                              |
       +----------------------------------------------------------------+
       |  1 Iopt  Upsilon(4S)                                           |
       |  2 Iopt  Upsilon(4S) -> B0 B0~ only                            |
       |  3 Iopt  Upsilon(4S) -> B+ B-  only                            |
       +----------------------------------------------------------------+
       |  4 kf    e+e- -> qq~ (JETSET)                                  |
       |  5 kf    e+e- -> gamma -> "onium" (JETSET call luonia)         |
       +----------------------------------------------------------------+
       | 10       tau+ tau- KORALB                                      |
       | 11       beam-gas EPC                                          |
       | 12       cosmic rays HemiCosm                                  |
       +----------------------------------------------------------------+

       DETAILS
       -------
       IMODE=-100 Do not call any physics generators.
                Useful if you want to look at TURTLE background
```

```
         by itself.

IMODE= -99 Call subroutine BEGUSER to generate the event.

IMODE=-10 to -13 same as IMODE 0 to -3 except:

         Pt (momentum transverse to beam pipe) instead of P
         momentum in GeV/c

IMODE=-3 same as IMODE=-2 except:

         x3 x4 = cos(The_min) cos(The_max)

         Angles in degrees, momentum in GeV/c

IMODE=-2 same as IMODE=0 except:

         x1 x2 = Pmin Pmax
               x3 x4 = The_min The_max
                     x5 x6 = Phi_min Phi_max

         A random value for P, theta and phi is chosen for each event
         in the ranges specified by x1-x6
         Angles in degrees, momentum in GeV/c

IMODE=-1 same as IMODE=0 except:

         x1 x2 x3 = P cos(theta) Phi
         Angles in degrees, Momentum in GeV/c

IMODE=0  User defined particle and four-vector. The particle is
         allowed to decay using JETSET.

         KF = JETSET code to identify particle

         x1 x2 x3 = P Theta Phi
                  x4 x5 x6 = ignored
         P  = momentum of particle in lab frame (GeV/c)
         Theta, Phi = direction of mom. vector in lab frame (degrees)

IMODE=1  Generate Upsilon(4S) at rest in cm frame. Decay using JETSET
         and boost to lab frame.

         Angular distribution of decay:
         Iopt = 0 (default) flat in cos(theta)
```

```
                Iopt = 1          sin^2(theta)

       IMODE=2  Same as IMODE=1 with Upsilon(4S) -> B0 B0~ decay only.

       IMODE=3  Same as IMODE=1 with Upsilon(4S) -> B+ B-  decay only.

       IMODE=4  Generate e+e- -> q q~ with JETSET, decay with JETSET.
                Get Ecm and boost from BEAM parameters.

                KF=0  Generate all quark pairs up to heaviest allowed by Ecm
                KF=n  Generate only quark pairs defined by n
                      n=1,2,3,4,5 = d,u,s,c,b

       IMODE=5  Generate e+e- -> gamma -> "onium" -> ggg or gg gamma
                                                  -> parton shower
                with JETSET, decay with JETSET.
                Get Ecm and boost from BEAM parameters.

                KF=0  Generate g g g events only
                KF=n  Generate mixture of ggg and gg gamma determined by
                      Q^2 of n
                      n=1,2,3,4,5 = d,u,s,c,b

       IMODE=10 Generate tau+ tau- using KORALB
       IMODE=11 Generate Beam-Gas interactions
                EPC library calculates the cross-sections
       IMODE=12 cosmic-ray generator, see data card COSM
```

CP: Select CP violation and/or B^0-$\bar{B}^0$ mixing

```
****************************************************************************
CP     Ichoice Par1 Par2 Par3 Par4 IPar       Select CP violation/B0 mixing
****************************************************************************
CP 0              Defaults
    Ichoice=0  no CP violation or B0 mixing
    Ichoice=1  use BBKING version of CP/mixing routine ASMXCP_BBKING
    Ichoice=2  use ASLUND version of CP/mixing routine ASMXCP_ASLUND

    Defaults for Ichoice=1 and 2 are different

CP 1 xd xs sincp coscp icp
     .7  8. .5    0.    1       Defaults
```

```
CP 2 sin_2alpha sin_2beta sin_2gamma coscp
      .5          .8       .5        0.       Defaults
```

BACK: Control beam background from TURTLE files

```
****************************************************************************
BACK  imode ix1 ix2 rmean         Control beam background from TURTLE files
****************************************************************************
BACK   0     0   0   0.    Defaults

       Add off-momentum beam particles and bremsstrahlung photons to
       the current event. The current implementation samples particles
       from a set of TURTLE files previously prepared.

       If no GEN data card is present then no physics generator will be
       called, i.e. the event will contain only background.

            imode   .ne.0  ==>  turn on background
                      = 1   end job at end of background input file
                      = 2   rewind file if EOF reached and reuse same
                               background events

             ix1,ix2   range of beam crossings over which to generate
                       backgrounds. (0 corresponds to trigger beam
                       crossing.)

             rmean     = 0.0  then use probability per beam crossing data
                              contained in above file to calculate the
                              number of events.
                       > 0.0  then input a fixed rmean number of events
                              spread over the beam crossing range ix1,ix2.
                              Use prob/crossing data to determine RELATIVE
                              sampling frequency from files.
                       < 0.0  then use a Poisson with mean -rmean to
                              determine the number of events to spread over
                              over the selected beam crossing range.
                              Use prob/crossing data to determine RELATIVE
                              sampling frequency from files.
```

BACF: Location of TURTLE file list

```
***************************************************************************
BACf 'filename'                                    Location of TURTLE file list
***************************************************************************
BACF ''              Defaults
      filename = Full pathname to a file containing a list of TURTLE files
                 and some information about out the files were generated.

For example file $BFROOT/data/ap75.list which contains:

2
 1.47000e-02  25      /nfs/juno/u5/data/heb.dat
 4.34200e-02  20      /nfs/juno/u5/data/leb.dat

The first number in the file is the number of files of TURTLE rays in
the list.  In this sample there are two files of turtle rays (one for
high energy beam (heb) and one for low (leb)).  The first column is the
probability per crossing for the events in this file and the second column
is the number of events contained in the file.

The format of the ray files differ in some important respects from the
previous OBJEGS input files and should not be considered compatible.
```

COSM: Control cosmic ray generator

```
***************************************************************************
                                               Control cosmic ray generator
COSM   FFuse   Pmode Pmin Pmax   Geant DetLen DetRad DetOff
***************************************************************************
COSM   TRUE    3     1.0  1000.0  TRUE  870.0  420.0  0.0 Defaults

       FFuse    Use FFREAD card (not local defaults in BegHemiCosm)
       Pmode    Momentum spectrum type
                  Pmode = 0  Constant = Pmin
                  Pmode = 1  1/p^2
                  Pmode = 2  From measured values
                  Pmode = 3  DarInv from theory, with cos(theta) dependence
       Pmin     Minimum p (GeV/c)
       Pmax     Maximum p (GeV/c)
       Geant    Use GEANT detector geometry (if present)
                  instead of the following
       DetLen   Detector cylinder length (cm)
       DetRad   Detector cylinder radius (cm)
       DetOff   Detector cylinder offset (cm)
```

```
example:
  1/p^2 spectrum:
    COSM TRUE 1

  p spectrum from files and use a large detector:
    COSM TRUE 2 1.0 1000.0 FALSE 12000.0 500.0 0.0 #

See the file src/hemicosm.F for more details,
or run one-event jobs and read the printouts.
```

B0B0: Selectively deactivate B^0 and/or $\bar{B}^0$ decay modes

```
****************************************************************************
B0B0   IB0  IB0~           Selectively deactivate B0 and/or B0~ decay modes
****************************************************************************
B0B0     1    1  Defaults
       IB0 = controls B0  decay
       IB0~= controls B0~ decay
       processed by subroutine BDECAY

example:           generates:
GEN    2           Upsilon(4S) -> B0 B0~
B0B0  11 3                        |  +---> mu- nu_mu~ X
                                  +------> Psi K0L
                                            +------> e+e- or mu+mu-

       For decay modes 11,12, and 13 the Psi decay modes are switched to
       e+e- and mu+mu- only. The user can override this with
       the JPSI data card (see below)

       IB0=0 turn off all decays
           1 allow all decays
           2 only B0 -> e   nu_e   X
           3         -> mu  nu_mu  X
           4         -> tau nu_tau X
           5 unused
           6 unused
           7 unused
           8 only B0 -> e X or mu X or tau X
           9         -> e X or mu X
          10 unused
```

```
11 only B0 -> Psi K0L   \
12         -> Psi K0S   | Psi -> e+e- or mu+mu-
13         -> Psi K*0   /  (see JPSI data card)
14         -> D+ D-
15         -> D*+ D*-
16         -> pi+ pi-
17         -> rho+/- pi-/+
18         -> a1+/-  pi-/+
```

JPSI: Control J/ψ decay modes

```
****************************************************************************
JPSI   n                                              Control J/Psi decay modes
****************************************************************************
JPSI  -1          Defaults
      n=-1        all decays allowed unless B0B0 selected decay 11,12,13
                                     then only e+e- and mu+mu- allowed
         0        turn off all decays
         1        all decays allowed
         2        Psi -> e+e- only
         3               mu+mu- only
         9               e+e- and mu+mu- only
```

JETD: Specify a file to read in JETSET decay information

```
****************************************************************************
JETD   'filename'        Specify a file to read in JETSET decay information
****************************************************************************
JETD   ''        Defaults
                 WARNING!!!!!
                 WARNING!!!!! You must use single quotes around the filename
                 WARNING!!!!! e.g. JETD 'myfile.dat'
                 WARNING!!!!!
     filename = Relative or absolute pathname that contains JETSET decay
                 information a la LUUPDA. This info overrides the built-in
                 decays. The file must contain the full decay table for a
                 given particle but does not need to redefine every particle.
                 ASLUND and BBKING users will recognize this as .newdat

      The LUCOMP mapping in the stand-alone and GEANT version is
        KF     LUCOMP(KF)
      -----    ----------
     70533      404          Upsilon(4S)
     80533      405          Upsilon(5S)
    nXXXXX      410+abs(n)   user definable particles

     See Section 2.3.1 concerning LUCOMP in ASLUND
```

BEAM: Set the beam energies

```
****************************************************************************
BEAM   Pz(electron) Pz(positron) Ecm   (GeV)          Set the beam energies
****************************************************************************
BEAM   9.0 -3.1 0.  Defaults
       Pz_electron, Pz_positron    z component of momentum of electron and
                                    positron beam (GeV). Px = Py = 0.
       Ecm                          center-of-mass energy in (GeV)

       You may set EITHER the two beam momenta OR one beam momentum and
       the center-of-mass energy. Set the unused quantity to zero and it
       will automatically be filled in with the correct value. Note that
       the sign of the beam momenta is important.
```

SBEAM: Set the beam energy spread

```
****************************************************************************
SBEAM  Sigma(electron) Sigma(positron) (GeV)      Set the beam energy spread
****************************************************************************
SBEAM   0.  0.   Defaults

The beam smearing is handled as in the native ASLUND generator, which
smears the nominal beam energy by a gaussian for every event.  For
each event the center-of-mass is calculated and the mass of the
Upsilon(4S) is set equal to this energy. The Upsilon(4S) is then
generated with zero momentum. The boost of the final state is also
based on the smeared energies on an event by event basis.

Warning, If the center-of-mass energy for a smeared event happens to
be below the mass threshold for B0 production an empty event will be
generated. The actual threshold in JETSET is slightly higher than
twice the B0 mass.
```

VERT: Set the primary vertex location

```
*****************************************************************************
VERT  x y z    (mm)                          Set the primary vertex location
*****************************************************************************
VERT 0. 0. 0.  Defaults

The given coordinates, in millimeters, sets the primary vertex position
```

SVERT: Set the primary vertex spread

```
*****************************************************************************
SVERT  dx dy dz (mm)                           Set the primary vertex spread
*****************************************************************************
SVERT 0. 0. 0.  Defaults
```

IVERT: Set the primary vertex smearing method

```
*****************************************************************************
IVERT Ix Iy Iz                        Set the primary vertex smearing method
*****************************************************************************
IVERT 0  0  0   Defaults
                0=smear with gaussian (sigma given by SVERT)
                1=smear with flat distribution (-delta,delta),
                         (delta given by SVERT
```

GENP: Print event information on the terminal

```
*****************************************************************************
GENP   IHEP ILUND IGPKINE IGPVERT   Print event information on the terminal
*****************************************************************************
GENP     0 0 0 0   Defaults
       IHEP >0   call heplist(IHEP)  after every event to list particles
       ILUND>0   call lulist(ILUND)  after every event to list particles
       IGPKINE>0 call GPKINE(0) after every event to list GEANT particles
       IGPVERT>0 call GPVERT(0) after every event to list GEANT vertices
                 IGPKINE and IGPVERT work only in GEANT programs

                 All four printouts can be activated simultaneously
```

JETO: Print the decay table for the listed particles

```
****************************************************************************
JETO KF1 KF2 ... KF          Print the decay table for the listed particles
****************************************************************************
JETO 0 0 ... 0    Defaults

               For any non-zero JETSET particle code the decay table for
               that particle will be listed at the end of the initialization
               pass through the JETSET generator.
example:
JETO 511 443   # list the decay table for B0 and J/PSI
```

RAN: Control the random number seeds

```
****************************************************************************
RAN Iseed1 Iseed2                                Control the random number seeds
****************************************************************************
RAN 1 0           Defaults

                  This data card initializes the random number seed for
                  the BEGET random number generator (RANECU). RAN behaves
                  exactly like the RNDM data card of GEANT.

                  The preselected random number seeds (see below)
                  result in distinct sequences 1 billion numbers apart.
                  Not all integer seeds result in a good or random
                  sequence, so the authors recommend using the preselected
                  ones.

      Iseed2=0    Iseed1 can be a number between 1 and 100 that causes a
                  preselected seed to initialize the random number generator.

      Iseed2<>0   The random number generator will be initialized with seeds
                  Iseed1 and Iseed2.
```

EVNT: Set the number of events to generate

```
****************************************************************************
EVNT Nbeg_events                            Set the number of events to generate
****************************************************************************
```

```
EVNT 1            Defaults

                  This data card merely sets a variable (Nbeg_events) in
                  the BEGET common block. It is up to the user to check it
                  and call the BEGET subroutine the appropriate number of
                  times.

                  ASLUND and BBSIM users should not use this card since event
                  counting is already built in to those programs.
```

REJE: Activate user event rejection function

```
****************************************************************************
REJE Ireje                          Activate user event rejection function
****************************************************************************
REJE 0            Defaults

                  This data card, in conjuction with the use function
                  BEGUREJECT, allows the user to generate specific events by
                  running a generator and then deciding whether the event
                  is interesting. This is most useful in the context of
                  sending events to GEANT, where simulation time is quite
                  long.

     Ireje = 1    Call user function BEGUREJECT. If the functions returns
                  true then generate another event call BEGUREJECT again.
                  The event is not passed on to the main program until
                  BEGUREJECT returns false.

     Ireje = 0    Do not reject any events.
```

6 STDHEP Common Block

The variables in the STDHEP common block are given in Table 5. The complete documentation including a table comparing the STDHEP particle codes with those of the Particle Data Group, JETSET, ISAJET, and HERWIG can be found in Reference [5].

Table 5: STDHEP common block variables.

NEVHEP	event number
NHEP	number of entries in the event
For each entry i	
ISTHEP(i)	status code for particle
IDHEP(i)	particle ID code
JMOHEP(1,i)	position of mother particle
JMOHEP(2,i)	position of second mother particle
JDAHEP(1,i)	position of first daughter
JDAHEP(2,i)	position of last daughter
PHEP(1-3,i)	p_x, p_y, p_z in (GeV/c)
PHEP(4,i)	Energy (GeV)
PHEP(5,i)	Mass (GeV/c^2)
VHEP(1-3,i)	x,y,z or vertex (mm)
VHEP(4,i)	production time (mm/c)

References

[1] *GEANT Detector Description and Simulation Tool,*, CERN Program Library W5103, CERN, March 1994.
`http://asis01.cern.ch/cn/CNASDOC/geant/GEANTMAIN.html`

[2] Originally written by Alan Weinstein, 1989.
T. Glanzman, W. Innes, S. Szena, and A. Snyder, *ASLUND: The Design of a Unix-based Group Software Project*, Ba$\bar{\rm B}$ar Note # 116, November 1993.
T. Glanzman, W. Innes, S. Szena, and A. Snyder, *Using ASLUND at SLAC*, Ba$\bar{\rm B}$ar Note # 117, November 1993.
`http://www.slac.stanford.edu/BFROOT/doc/PhysSim/aslund.txt`

[3] T. Sjöstrand, *JETSET 7.3*, CERN-TH 6488/92, May 1992.

[4] S. Jadach and Z. Was, *KORALB 2.1; An Upgrade with Tauola Library of Tau Decays*, Comput. Phys. Comm. **64**, 267(1991).

[5] Lynn Garren, *STDHEP 1.05*, FNAL PM0091, September, 1993.
`http://www.slac.stanford.edu/BFROOT/doc/Miscellaneous/stdhep.ps`

[6] *FFREAD Format Free Input Processing*, CERN Program Library I302,
`http://asis01.cern.ch/cn/CNASDOC/WWW/ffread/ffmain/ffmain.html`

[7] *Letter of Intent for the Study of CP Violation and Heavy Flavor Physics at PEP-II*, BaBar Collaboration, SLAC-442, June 1994.
`http://www.slac.stanford.edu/BFROOT/doc/Adminsitration/LOI`

[8] *HBOOK Reference Manual*, CERN Program Library Y250,
`http://asis01.cern.ch/cn/CNASDOC/hbook/HBOOKMAIN.html`